AMENDMENTS TO THE CLAIMS

Claims 1, 11, and 19 are currently amended. Claims 5, 17, and 23 are canceled. Claims 2-4, 6-10, 12-16, 18, 20-22, and 24-28.

1. (currently amended) A method comprising:

aligning an address stored in a K-bit word having N least significant bits such that the N least significant bits are zero; and

encoding information in N bits; and

storing the encoded information in the N least significant bits of the K-bit word, wherein the encoded information about the object comprises at least one of hash code information, lock information, garbage collection information, and reflection information.

2. (original) The method according to claim 1, wherein the address is a pointer to where runtime properties of an object are stored in memory.

3. (original) The method according to claim 1, wherein the address is a pointer to at least one of metadata and a table of virtual functions within a class.

4. (original) The method according to claim 1, wherein the encoded information is information about an object.

5. (canceled)

6 (original) The method according to claim 1, wherein aligning comprises:

allocating a memory space to store the runtime properties of an object;

storing the runtime properties of the object at an address within the memory space such that the N least significant bits of the address are zero; and

storing the address in the K-bit word.

7. (original) The method according to claim 6, wherein the memory space is larger than an amount of space necessary to store the runtime properties of the object.

8. (original) The method according to claim 1, wherein N is equal to eight.

9. (original) The method according to claim 1, further comprising:

encoding lock information into at least two of said N least significant bits.

10. (original) The method according to claim 9, said encoding lock information comprising:

storing a zero in the a bit of a two-bit lock code and a zero in a second bit of said two-bit lock code if an associated object is not locked;

storing a one in the first bit of said two-bit lock code and a zero in the second bit of said two-bit lock code if an associated object is locked by one thread; and

storing a one in the first bit of said two-bit lock code and a one in the second bit of said two-bit lock code if an associated object is at least one of recursively locked by the same thread and locked by multiple threads.

11. (currently amended) A device comprising:

first and second addressable units, the first addressable unit storing a K-bit word, the K-bit word having N least significant bits and including

an address of the second addressable unit, wherein the address of the second addressable unit is an address at which the N least significant bits are equal to zero, and

encoding information in N bits; and

storing the encoded information in the N least significant bits of the K-bit word, wherein the encoded information about the object comprises at least one of hash code information, lock information, garbage collection information, and reflection information.

12. (original) The device according to claim 11, wherein the K-bit word is a pointer to the second addressable unit.

13. (original) The device according to claim 11, wherein the second addressable unit comprises least one of metadata and a table of virtual functions within a class.

14. (original) The device according to claim 11, wherein the encoded information is information about an object.

15. (canceled).

16. (original) The device according to claim 11, wherein N is equal to eight.

17. (original) The device according to claim 11, wherein the information includes a a lock code comprising at least two bits.

18. (original) The device according to claim 17, wherein said lock code is a two-bit lock code, and wherein the two-bit lock code indicates one of a free lock, an easy lock, and a heavy lock.

19. (currently amended) A machine accessible medium containing program instructions that, when executed by a processor, cause the processor to:

align an address stored in a K-bit word having N least significant bits such that the N least significant bits are zero; and

encode information in N bits; and

store the encoded information in the N least significant bits of the K-bit word, wherein the encoded information about the object comprises at least one of hash code information, lock information, garbage collection information, and reflection information.

20. (original) The machine accessible medium according to claim 19, wherein the address is a pointer to where runtime properties of an object are stored in memory.

21. (original) The machine accessible medium according to claim 19, wherein the address is a pointer to at least one of metadata and a table of virtual functions within a class.

22. (original) The machine accessible medium according to claim 19, wherein the encoded information is information about an object.

23. (canceled)

24. (original) The machine accessible medium according to claim 19, further comprising instructions that, when executed by a processor, cause the processor to:

allocate a memory space to store the runtime properties of an object;

store the runtime properties of the object at an address within the memory space such that the N least significant bits of the address are zero; and

store the address in the K-bit word.

25. (original) The machine accessible medium according to claim 24, wherein the memory space is larger than an amount of space necessary to store the runtime properties of the object.

26. (original) The machine accessible medium according to claim 19, wherein N is equal to eight.

27. (original) The machine accessible medium according to claim 19, further comprising instructions that, when executed by a processor, cause the processor to:

encode lock information into at least two of the N least significant bits.

28. (original) The machine accessible medium according to claim 19, further comprising instructions that, when executed by a processor, cause the processor to:

store a zero in a first bit of a two-bit lock code and a zero in a second bit of said two-bit lock code if an associated object is not locked;

store a one in the first bit of said two-bit lock code and a zero in the second bit of said two-bit lock code if an associated object is locked by one thread; and

store a one in the first bit of said two-bit lock code and a one in the second bit of said two-bit lock code if an associated object is at least one of recursively locked by the same thread and locked by multiple threads.